УДК 681.3

СИСТЕМНАЯ ОБОЛОЧКА BRAINSTORM ДЛЯ ПОДДЕРЖКИ УНАСЛЕДОВАННОГО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

В.П. Зимин, Е.А. Хомяков

Томский политехнический университет
E-mail: zimin@ido.tpu.ru

Анализируется проблема поддержки унаследованного программного обеспечения, возникающая при сопровождении существующих пакетов прикладных программ. Предлагается решение данной проблемы с помощью системной оболочки BrainStorm v. 1.0 под MS Windows, которая позволяет создавать и сопровождать пакеты программ, ранее исполняющиеся под MS-DOS. Описываются основные механизмы системного обеспечения, реализованные в BrainStorm. Приведены результаты тестирования данного инструментального средства.

За последние несколько десятков лет, вследствие интенсивного развития информационных технологий и математического моделирования, было создано большое количество универсальных и специализированных проблемно-ориентированных пакетов прикладных программ (ППП). Создание таких пакетов является актуальной задачей, так как они фактически являются одной из форм поддержки и передачи знаний и технологий. Но объективные и субъективные трудности разработки, создания и сопровождения таких комплексов программ приводят к ряду проблем, одной из которых является проблема унаследованного программного обеспечения (ПО). Суть данной проблемы состоит в том, что в силу ряда причин разработанные программы перестают эксплуатироваться, поэтому новым поколениям исследователей приходится вновь тратить значительные ресурсы на создание программных пакетов с подобными функциями. Эти причины можно классифицировать как субъективные и объективные. Субъективные причины обусловлены, прежде всего, недостаточным уровнем поддержки программного обеспечения пакетов, в связи с распадом коллективов разработчиков, частичной или полной утратой исходных текстов программ. Объективные причины появления унаследованного ПО обусловлены самой методологией создания ППП. Рассмотрим подробнее объективную составляющую проблемы унаследованного ПО.

Методология разработки проблемно-ориентированных программных комплексов рекомендует разделять программное обеспечение пакетов на функциональную и системную части [1, 2]. Функциональная часть ППП содержит программную реализацию вычислительных алгоритмов для исследуемой предметной области, а системная часть – программную поддержку разнообразных спецификаций взаимодействия и сервисы: интерфейс и технологию диалога пользователя с пакетом, межпрограммные интерфейсы, общие алгоритмы обработки данных (в том числе визуализацию данных) и т.п. Одной из важных характеристик ПО является его жизненный цикл (ЖЦ), который представляет собой совокупность этапов (временных интервалов) разработки, эксплуатации и сопровождения созданных программных средств. На ранних стадиях развития как программного и аппаратного обеспечения ЭВМ в целом, так и функционального и системного обеспечения ППП временной интервал ЖЦ последних фактически совпадал. Это обстоятельство, в частности, объяснялось слабым развитием системного обеспечения ЭВМ: ввод данных с перфокарт и перфолент, текстоориентированный диалог без дополнительных проверок вводимых данных, вывод в виде массива чисел, в лучшем случае – вывод графиков на бумажный носитель. С появлением персональных компьютеров (ПК) ситуация в развитии системного обеспечения существенно изменилась: появился полнотекстовый, а затем и графический интерфейс пользователя, диалог в виде меню, возможность распределенной, удаленной обработки данных и другие технологии создания, эксплуатации и сопровождения ПО. Смена технологий создания системного ПО происходила и происходит стремительно, за короткий промежуток времени. Например, в [3] предлагается расширенное толкование закона Мура, распространение его на системное обеспечение компьютеров: смена операционной системы каждые 3 года, изменение интерфейсов каждые 6 лет. В тоже время функциональное наполнение пакетов, которое в основном определяется вычислительными алгоритмами, меняется гораздо медленнее. Хотя и здесь появились и развиваются новые технологии обработки данных. Уместно отметить такое новое направление развития функционального обеспечения ППП как алгоритмы параллельных высокопроизводительных вычислений на вычислительных системах В нашей стране многопроцессорные вычислительные системы мало распространены в силу их дороговизны, хотя в последние годы это направление развития ПО стимулируется появлением сравнительно дешевых кластеров ПК.

В силу разной скорости изменения (обновления) системного и функционального наполнения ППП проявляется объективная проблема поддержки унаследованного ПО. С одной стороны, системное ПО перестает поддерживаться в должной мере на уровне операционной системы, а с другой стороны, функциональное ПО пакетов должно постоянно модернизироваться для синхронизации с жизненном циклом системного наполнения ППП.

Причем решение данной проблемы существенно зависит от той системной информационной технологии, которую необходимо поддерживать, для получения новых функциональных качеств программных комплексов. Например, в [4] рассматривалось решение проблемы унаследованного ПО с точки зрения обеспечения удаленного доступа к вычислительным ресурсам. В настоящей статье рассматривается решение данной проблемы с точки зрения поддержки функционального обеспечения, разработанного для эксплуатации в операционной системе (ОС) MS-DOS, под ОС MS Windows, и предоставление пользователям соответствующих сервисов. В основе решения проблемы лежит идея использования при создании системного обеспечения ППП минимального набор механизмов (технологий) интеграции функционального наполнения пакетов, которые с течением времени не изменяются, или меняются сравнительно мало.

С целью определения минимальных требований к инструментальному средству поддержки унаследованного ПО, проанализируем некоторые тенденции в методологии создания ППП, а так же состояние системного и функционального обеспечения ранее разработанных пакетов. При решении сложных научных и инженерно-технических проблем существенно повышается значение математического моделирования. Однако аналитические возможности последнего ограничиваются, фактически, классом линейных задач. Решение актуальных задач науки и техники требует применение ЭВМ и технологии вычислительного эксперимента (ВЭ) и, как следствие, разработку и эксплуатацию ППП – средство программного обеспечения ВЭ [1]. Подобные проблемы возникают и при создании и проектирования сложных образцов техники и технологии, когда требуется проводить математическое моделирование на всех этапах их проектирования и испытания. В [5] отмечается, что в разрабатываемом ППП должно быть представлено три уровня описания предметной области:

1) Этап разработки технического задания и технических предложений. На этом этапе в основном выполняются оперативные расчеты по упрощенным моделям.
2) Этап эскизного проектирования. Используются апробированные модели среднего уровня сложности, но учитывающие распределенность объекта (процесса) проектирования.
3) Этап проведения НИР, получение новых фундаментальных знаний о работе конструкций. Создание новых сложных математических моделей, их апробация.

При решении как научных, так и инженерно-конструкторских задач требуется использовать технологию ВЭ, обеспечивая, в первую очередь, требования многомодельности и многовариантности при проведении вычислительных работ. В последующих работах [6, 7] перечень этапов технологии создания ППП фактически повторяет этапы ВЭ, но вместе с тем происходит более углубленное толкование каждого из этапов, обобщение опыта создания конкретных ППП.

Анализ имеющихся пакетов и средств их построения позволил выявить следующие закономерности в их развитии [1, 2, 5–8].

1) Организация входных/выходных и промежуточных данных функциональных модулей:
   - различного типа файлы;
   - банки данных;
   - различные БД (которые так же являются файлами со специальной структурой);
   - хранилища данных;
   - иное, например, хранение неструктурированных данных в различных форматах.
2) Оформление логических единиц функционального обеспечения пакетов:
   - модули в виде исполняемых или интерпретируемых программ с произвольным интерфейсом;
   - модули в виде исполняемых или интерпретируемых программ со специализированным интерфейсом;
   - модули-процедуры.
3) Объединение логических единиц функционального обеспечения ППП:
   - с помощью специализированного языка.
4) Организация работы пользователя с помощью различных элементов пакета:
   - входных/выходных данных для функционального наполнения;
   - логических единиц функционального наполнения при их запуске;
   - истории ведения исследований различного уровня (научных, проектных, оценочных);
   - интерфейса для организации диалога пользователя с пакетом (текстоориентированный, графический интерфейс; формирование заданий, пакетов и т.п.).

Таким образом, с точки зрения пользователя при создании инструментального средства создания и поддержки унаследованного ПО необходимо сохранить технологию вычислительного эксперимента, обратив основное внимание на организацию взаимодействия функциональных модулей и входных/выходных данных.

С самых общих позиций функциональный модуль можно представить как программу, обрабатывающую по заданному алгоритму входные данные и формирующую по определенным правилам выходные данные. С учетом этих обстоятельств, проведенный анализ системного обеспечения существующих ППП позволил сформулировать функциональные требования, которым должно удовлетворять создаваемое инструментальное программное средство:

1) Оперативная классификация (разбиение на классы) множества функциональных модулей ППП.
2) Работа с разными совокупностями данных, с разными банками данных.
3) Быстрый запуск функциональных модулей ППП, использование предопределенных пользователем файлов ввода/вывода, которые описывают конфигурацию входных/выходных данных функциональных модулей.
4) Создание исполняемых (линейных) цепочек из функциональных модулей.
5) Ведение редактируемого журнала результатов запуска функциональных модулей и цепочек модулей.
6) Реализация современного интерфейса системной оболочки (Window-интерфейса).
7) Визуализация входных/выходных данных функциональных модулей на плоскости.

Общей составляющей практически во всех перечисленных выше требованиях к системной оболочке является задача эффективной организации обмена данными. Анализ общих механизмов операционных систем типа MS-DOS, Win16 и Win32 позволил выбрать наиболее простую и надежную модель обмена данными: использование системы управления файлами и командную строку. Оба механизма взаимодействия программ и данных имеются практически во всех операционных системах, причем их использование не требует дополнительного анализа и структурирования потоков данных. Отметим, что как Win16, так и Win32 имеют значительно больше возможностей по организации взаимодействия программ, чем ОС MS-DOS.

Рассмотрим подробнее реализацию выше указанных требований в рамках системной оболочки BrainStorm, базовое окно которой представлено на рис. 1.

Для целей оперативной классификации функциональных модулей пользователем организуется требуемая иерархическая структура папок (категорий), с соответствующими названиями. С помощью технологии drag&drop из папки «Все модули» (где физически находится все множество функциональных модулей) осуществляется наполнение папок категорий, при этом используется механизм ссылок на модули.

Многовариантность вычислительного эксперимента отображается в виде множественности входных/выходных данных функциональных модулей. Задача сохранения вариантных численных экспериментов решается двумя способами. Первый способ состоит в дублировании имен файлов входных/выходных данных, но вариацию их расширений. Самый простой способ кодирования при этом – номер варианта эксперимента совпадает с трехзначным числом, полученным из цифр расширения имени файла. Второй способ – организация банков данных с помощью системной оболочки (рис. 1, «Банки данных»). Банки данных, наборы входных/выходных данных функциональных модулей, организуются пользователем. При работе с конкретным банком данных требуется определить его как «Текущий». При большом количестве вариантных расчетов второй способ является предпочтительным.

Для интеграции функционального модуля, написанного на языке Turbo Pascal 7.0, в системную оболочку используется следующая технология. Входные и выходные данные модуля объединяются в файлы ввода (`inputs.txt`) и вывода (`out-`

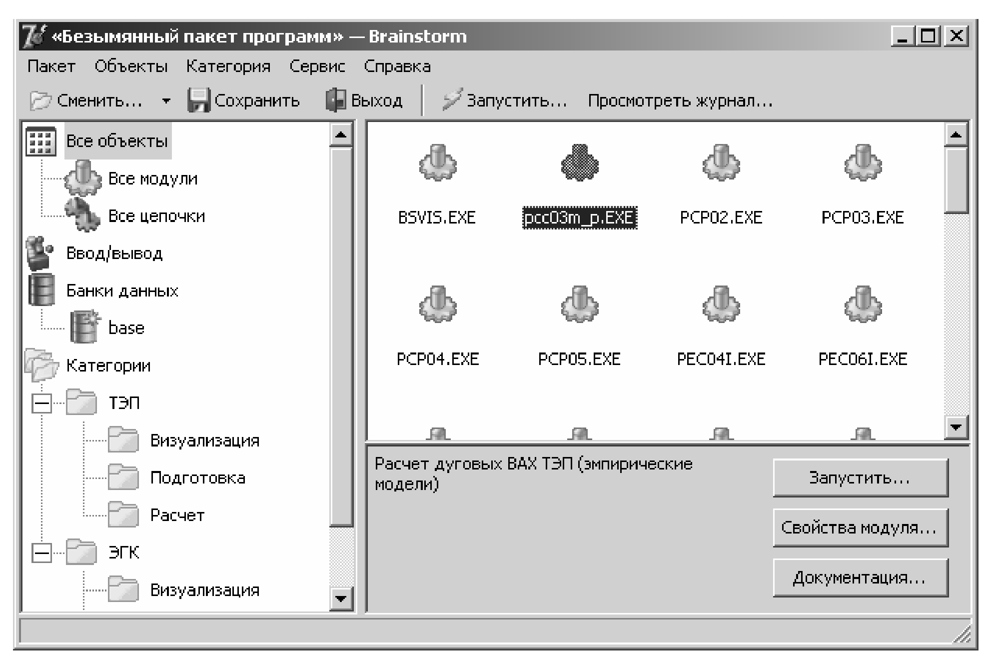


Рис. 1. Базовое окно системной оболочки BrainStorm с загруженным ППП

puts.txt) (рис. 1, папка «Ввод/вывод»), которые получает модуль при запуске от системной оболочки. Кроме этого, указывается режим выполнения модуля: пакетный или интерактивный (ключ batch). В пакетном режиме весь диалог в модуле подавляется. Например, запуск модуля pcc03m_p.exe в среде системной оболочки BrainStorm в пакетном режиме будет выглядеть таким образом:

```
pcc03m_p.exe --bs-i=inputs.txt
--bs-o=outputs.txt --bs-batch.
```

Структуры текстовых файлов ввода и вывода модуля идентичны и состоят из строк, содержащих описание вида «ключ=значение». Например,

```
/ptcscd=..\banks\base\Ptcscd.001
/krekcd=..\banks\base\krekcd.011
/vndvcd=..\banks\base\VNDVCD.001
/dugjvcd=..\banks\base\Dugjvcd.050
```

В данном файле ввода в строках указаны имена файлов, которые выбираются из основного банка данных (..\banks\base\) и в которых содержаться, в конечном итоге, входные/выходные данные функционального модуля. В качестве таких данных могут быть указаны простые (текстовые) переменные, например:

```
Progress=1
Iteration=150
```

Такие переменные интерпретируются в выполняемом модуле. Для определения переменных как входных данных в теле модуля используется функция GetParam(param:string):string, для выходных данных – процедура PutParam(param,value:string). Для подавления диалога в пакетном режиме выполнения модуля используется глобальная логическая переменная BatchMode. Отметим, что файл вывода создается модулем (процедура SaveOutputFile;), что является индикатором успешного его выполнения. В системной оболочке сопровождаются описанием как функциональные модули (например, pcc03m_p.exe.desc), так и содержание файлов ввода (pcc03m_p.exe.inputs) и вывода (proba.exe.output). Например, состав файла ввода pcc03m_p.exe.inputs модуля pcc03m_p.exe следующий:

```
/ptcscd=Зависимость давления цезия от температуры резервуара с цезием
/krekcd=Работы выхода эмиттера и коллектора для электродной пары
…
```

На рис. 2 представлено окно запуска модуля pcc03m_p.exe. Реализуется пакетный режим выполнения модуля и указаны конкретные имена входного файла. Маршрут не представлен, т.к. установлен маршрут текущего банка данных – ..\banks\base\ (рис. 1).

Системная оболочка BrainStorm позволяет формировать из множества функциональных модулей в интерактивном режиме структуру линейных цепочек модулей. Пользователь указывает желаемую последовательность выполнения в пакетном режиме модулей и контролирует на основе файлов описания модуля и его файлов ввода/вывода (рис. 3). При этом формируется «эстафета» данных, которая показывает, какие данные (идентификаторы данных) пользователь будет иметь перед выполнением конкретного модуля цепочки и какие – после выполнения.

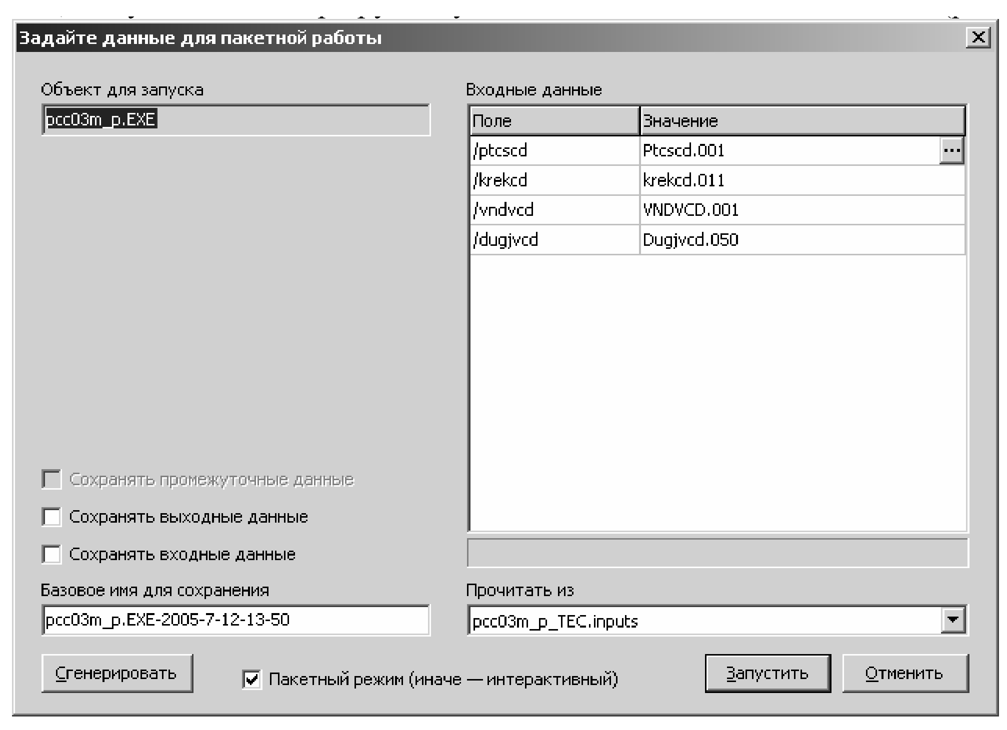


Рис. 2. *Окно запуска функционального модуля*

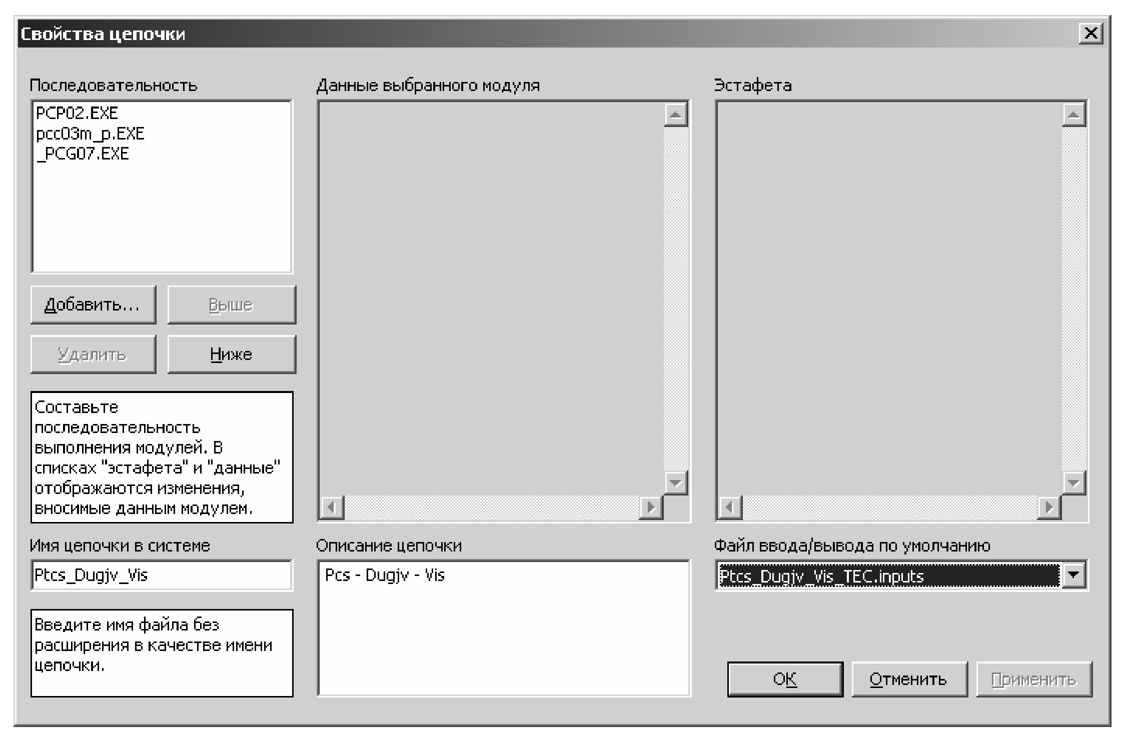


Рис. 3. *Окно для формирования цепочки модулей и контроля ее свойств*

Каждый запуск модуля или цепочки модулей протоколируется в журнале системной оболочке. Журнал – это редактируемый текстовый файл, в котором системная оболочка указывает имя исполняемого объекта (модуля или цепочки модулей), содержание файлов ввода/вывода и результат выполнения. Пользователь может добавлять в журнал свои комментарии.

Модули визуализации формируют файлы визуализации с XML-структурой, которые интерпретируются программой визуализации `bsvis.exe`.

Таким образом, использование сервиса системы управления файлами и механизма командной строки позволяет, с одной стороны, минимально модифицировать код программ, написанных под MS-DOS и включаемых в ППП, а с другой стороны – получить пользователю созданного пакета, поддерживаемого системной оболочкой BrainStorm, интерфейс под ОС MS Windows.

Первая версия системной оболочки BrainStorm написана с использование системы визуального программирования Delphi 7. Ядро оболочки представляет независимый модуль Tpackage, управляющий ППП. Управление пакетом осуществляется с помощью методов объекта: управление функциональными модулями, управление категориями (для классификации модулей), управление цепочками модулей, управление файлами ввода/вывода, управления банками данных, управление записями в журнал и т.д. После подготовки функциональных модулей для интеграции в ППП, с помощью BrainStorm создается специальная файловая структура, которая по мере ее наполнения отображается в интерфейсе системной оболочки.

Тестирование системной оболочки BrainStorm v. 1.0 проводилось двумя способами.

Во-первых, были реализованы функции по поддержки ППП с помощью моделирующей системы (системной оболочки) КОРТЕS [8], которая была написана на языке Object Pascal. Кроме интерфейса под ОС MS Windows, существенно расширены функции работы с функциональными модулями, с входными/выходными данными модулей, функции визуализации данных. Появились функции, позволяющие создавать и выполнять цепочки функциональных модулей. Тестирование системной оболочки BrainStorm полностью подтвердило выполнение функций адекватно функциям моделирующей системы КОРТЕS.

Во-вторых, был создан пакет программ KIAE_2004 для моделирования процессов в термоэмиссионных системах [9]. В пакете 31 функциональный модуль, написанный на языке Turbo Pascal v. 7.0. Банк данных пакета KIAE_2004 содержит более 100 входных/выходных файлов модулей. Структура и содержание данных при интеграции в пакет не изменялись, за исключением функциональных модулей визуализации.

При тестировании пакета KIAE_2004 были получены результаты моделирования, идентичные данным [8], а также новые результаты моделирования, изложенные в [9].

В заключении можно сделать следующие выводы:

- Для решения проблемы поддержки унаследованного программного обеспечения необходимо использовать основные механизмы эффективного обмена данными, характерные для опе-

рационных систем типа MS-DOS, а также Win16 и Win32: системы управления файлами и командную строку. Оба механизма взаимодействия программ и данных имеются практически во всех операционных системах, причем их использование не требует дополнительного анализа и структурирования потоков данных.

- Механизмы эффективного обмена данными в ППП реализованы в системной оболочки BrainStorm v. 1.0 под MS Windows и проверены при создании двух пакетов прикладных программ.
- Данную системную оболочку можно применять на этапе начального формирования (прототипирования) ППП, для написания функциональных модулей которого применяются современные системы программирования, когда более важным является сам факт передачи данных между модулями ППП, а не эффективность коммуникаций модулей.

СПИСОК ЛИТЕРАТУРЫ

1. Самарский А.А. Пакеты прикладных программ как средство обеспечения сложных физических расчетов // Перспективы системного и теоретического программирования: Труды Всес. симп.; Под ред. И.В. Потосина. – Новосибирск, 1979. – С. 5–13.
2. Яненко Н.Н., Карначук В.И., Коновалов А.Н. Проблемы математической технологии // Численные методы механики сплошной среды. – Новосибирск, 1977. – Т. 8. – Вып. 3. – С. 129–157.
3. Ильин Н.П. Расширенное толкование закона Мура // Информационные технологии. – 2005. – № 5. – С. 70–71.
4. Массель Л.В., Горнов А.Ю., Подкаменный Д.В. Создание вычислительных ресурсов в Internet на основе унаследованного программного обеспечения // Совместный выпуск журналов «Вычислительные технологии». – Т. 7 и «Вестник КазНУ». – № 4 (32). – Матер. Междунар. конф. «Вычислительные технологии и математическое моделирование в науке, технике и образовании». – Ч. 3. – Новосибирск-Алматы. – 2002. – С. 247–252.
5. О технологии изготовления инженерно-физических ППП. Пакет «Аркан» / С.С. Бондарчук, А.Б. Ворожцов, А.С. Жуков и др. // Языки и параллельные ЭВМ. – М.: Наука, 1990. – С. 79–90.
6. Потапчук Г.А. Инструментальные средства поддержки процесса создания программных систем дискретной оптимизации // Кибернетика и системный анализ. – 1992. – № 4. – С. 125–133.
7. Парасюк И.Н. Принципы разработки современных программных систем анализа данных // Кибернетика и системный анализ. – 1993. – № 3. – С. 155–162.
8. Бабушкин Ю.В., Зимин В.П., Синявский В.В. Моделирующая система KOPTES для исследования тепловых и электрических процессов в термоэмиссионных системах преобразования энергии // Ракетно-космическая техника: Труды. Сер. XII. – Вып. 1–2. Расчет, проектирование, конструирование и испытания космических систем / Под ред. В.В. Синявского. – Калининград, Моск. обл.: РКК «Энергия», ОНТИ, 1998. – С. 60–78.
9. Бабушкин Ю.В., Зимин В.П., Хомяков Е.А. Программное обеспечение и результаты моделирования термоэмиссионных систем // Известия Томского политехнического университета. – 2006. – Т. 309. – № 3. – С. 53–57.

УДК 543:681.3.066